# Welcome to the HTML Syntax Problem Set!

The purpose of this problem set is to help you practice writing HTML. We want you to build confidence and master basic HTML syntax.

If you get stuck, remember that there's no harm in going back and rewatching any part of the lesson. And we encourage you to use outside resources too! You can always refer to [**MDN references**](https://developer.mozilla.org/en-US/docs/Web/HTML/Element) or your favorite search engine to get unstuck.

You can do this! Have fun!

# Make a Button

For this quiz, all I want you to do is type out the HTML to create a button element.

[**A BUTTON**](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/73065685430923)

Buttons may look different based on browser and operating system.

The purpose of this quiz is to make sure that you're using the correct syntax. As you will realize when you build websites, you can write HTML incorrectly — by forgetting a closing tag, for example — and the browser will try to correct your mistake. Sometimes the browser will guess correctly and your website will look fine. But most of the time, it will be wrong and strange bugs will start appearing.

This is a good time to double check that you're creating elements correctly because you'll be writing more complicated HTML soon!

You create button elements the same way you've been creating other elements: <tag>content</tag>. In this case, the tag name is button and the content that comes between the tags will be the text displayed inside the button. Here's more about buttons on the [**Mozilla Developer Network (MDN)**](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/button).

When you create your button, make sure you include some text content so that the button says something — it can be whatever you want!

If you want to see what your button looks like (you don't have to for this quiz), simply create an HTML file with a button element and open it in your browser!

<button>TEST ME</button>

Here's my solution:

<button>a button!</button>

It follows the pattern: <tag>content</tag>. Here, the tag is button and the content turns into the button's text! If you tried opening this HTML in your browser, you should have made [**a button that looks like this**](http://udacity.github.io/fend/fend-refresh/lesson2/problem-set/button-example.html).

Click "Next" to move on to the next part of the problem set

# Make All the Headers

Websites have different ways to show you, the user, what is important. Take for example...

## Big Important Words!

This style of text **Big Important Words!** is called a **header**. This should look familiar because almost every website uses headers.

The New York Times mobile site uses them. Check out all of the images below. The heading has a red square around each heading.

**[[](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/90c4e4ec-36f2-4beb-a22d-3c7555b89db8)](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/90c4e4ec-36f2-4beb-a22d-3c7555b89db8)**

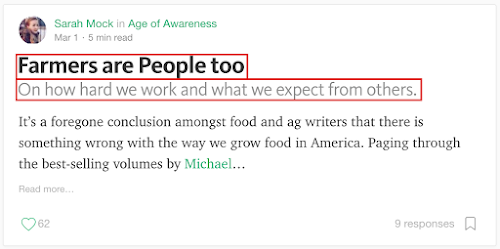
([***Read the sad story of Mr. Tomlinson's passing***](http://mobile.nytimes.com/2016/03/08/technology/raymond-tomlinson-email-obituary.html?_r=0))

**[[](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/90c4e4ec-36f2-4beb-a22d-3c7555b89db8)](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/90c4e4ec-36f2-4beb-a22d-3c7555b89db8)**

**[CNN sometimes adds headers to images](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/90c4e4ec-36f2-4beb-a22d-3c7555b89db8)**

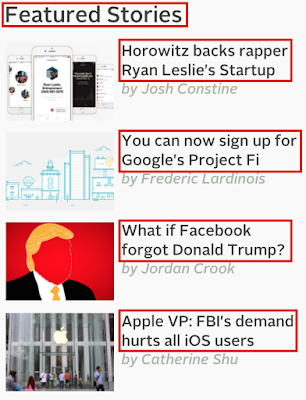
### Main Headings and Subheadings

It's also pretty common to see more than one kind of header being used. Here's an example of two headers on Medium.

**[[](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/90c4e4ec-36f2-4beb-a22d-3c7555b89db8)](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/90c4e4ec-36f2-4beb-a22d-3c7555b89db8)**

**[The main heading with "Farmers are People too", with a sub heading underneath it.](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/90c4e4ec-36f2-4beb-a22d-3c7555b89db8)**

TechCrunch using headers to show the titles of featured stories:

**[[](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/90c4e4ec-36f2-4beb-a22d-3c7555b89db8)](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/90c4e4ec-36f2-4beb-a22d-3c7555b89db8)**

**[A main heading of "Featured Stories" with each story having its own subheading.](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/90c4e4ec-36f2-4beb-a22d-3c7555b89db8)**

Let's experiment with headers in a real HTML document

## How to Complete this Exercise

See look, I just used a header to help you figure out why this section is here :)

1. Look at the Workspace on this page. You'll find index.html inside.
2. Edit index.html in the workspace and watch your website change in the preview panel.

**Make a List**

Did you know that web developers spend 67.7493% of their time looking things up?

Ok, I made up that number.

But seriously, making sense of documentation and looking up new techniques and technologies is a huge part of any web developer's work. And that's what you're going to do in this exercise.

For this exercise:

* Notice that what you're reading right now is an unordered list :) An unordered list usually displays with bullet points.
* Use the [**Mozilla Developer Network (MDN)**](https://developer.mozilla.org/en-US/docs/Web/HTML/Element) to research **unordered lists** (<ul>).
* Make an unordered list with the three web languages:
  + HTML
  + CSS
  + JavaScript

There are two types of lists - ordered lists (with numbers or letters) and unordered lists. Both of them require two kinds of elements. One is going to be nested inside the other (there will be one parent and multiple children).

**How to Complete the Exercise**

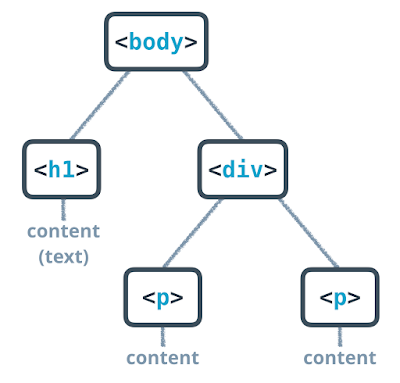
Look at the Workspace on this page. You'll find index.html inside. Edit index.htmlin the workspace and watch your website change in the preview panel.

I want you to: (*Look! Another unordered list!*)

* Add an unordered list element with three child elements (one each for HTML, CSS, and JavaScript).

# Tree to HTML

You've been learning about the relationship between data trees and HTML. Here's a sample tree for this exercise:

**[[](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/2f810ece-55db-4d1d-a029-e82bd88afbfb)](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/2f810ece-55db-4d1d-a029-e82bd88afbfb)**

**[The tree you're going to model](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/2f810ece-55db-4d1d-a029-e82bd88afbfb)**

## How to Complete this Quiz

You're going to build the HTML for the image, above. Take a close look at the relationships in the tree. Pay attention to parents, children, and siblings. Notice that some elements have text content.

Look at the Workspace, below. You'll find the file index.html inside. Edit index.html in the workspace and watch your website change in the preview panel.

For this exercise, I want you to:

* Add an <h1> tag with some content.
* Add a <div> tag.
* Add two paragraph tags as child elements to the<div>.
* Add text content to each paragraph element.

## Hyperlinks

The power of the web is that pages can lead to other pages. When you click on a link on a web page, it takes you to another page. This link is called a **hyperlink**.

Hyperlinks are created with [**anchor elements**](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/a), which generally look like:

<a href="https://www.udacity.com">Udacity</a>

and render on the page like this: [**Udacity**](https://www.udacity.com/).

Inside the opening a tag there is href, which stands for "reference." This is called an **attribute**. Attributes like href describe the properties of HTML elements. In this case, the href attribute is the target URL that the link will open. The content inside the anchor element is the text that users see displayed on the page.

This is the format that you must use when you make hyperlinks! Note:

* There is a space between a and href.
* There are no spaces around the =.
* The website has two " around it.
* There are no spaces between the href attribute and the > of the opening tag.

## How to Complete this Quiz

For this quiz, I want you to make your first hyperlink! On the next quiz, I'll give you a URL and the text that needs to be displayed. It will be your job to write the link!

<a href="https://www.google.com">Google</a>

## Add an Image

**[[](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/46ec0fa5-e3ac-435d-b354-4b837bf36ca7)](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/46ec0fa5-e3ac-435d-b354-4b837bf36ca7)**

[**Puppy image by dbking**](https://commons.wikimedia.org/wiki/File%3ASt._Bernard_puppy.jpg) (originally posted to Flickr as 272\_7242) [**CC BY 2.0**](http://creativecommons.org/licenses/by/2.0), via Wikimedia Commons

[**Kitten image by Steve-h**](https://commons.wikimedia.org/wiki/File%3AGreen_eyes_kitten.jpg) (Flickr: Maggie) [**CC BY-SA 2.0**](http://creativecommons.org/licenses/by-sa/2.0), via Wikimedia Commons

[**Earth image by Norman Kuring**](http://visibleearth.nasa.gov/view.php?id=78314), NASA GSFC, using data from the VIIRS instrument aboard Suomi NPP.

IMAGES! Images on the web are awesome. Time to make one.

For this exercise, you'll be linking an image in your Workspace. I'll give you the URL of an image and it will be your job to make it display.

An image is made with an <img> element. It looks like so:

<img src="http://somewebsite.com/image.jpg" alt="short description">

The source attribute, src, is like the href of a link - it is the URL of the image you want to display. For now, your images will need to be hosted online, which means that the URL will need to start with http:// or https://. You'll learn about another way to set image source in the next exercise. The alt attribute stands for "alternative description," which is important for people who use screen readers to browse the web. This is text that will show up in lieu of the actual image.

An image element is a little different than the elements you've seen before. Images do not need closing tags! (For the eager, these are called "[**void elements**](https://www.w3.org/TR/html52/syntax.html#void-elements)".)

Here's the [**MDN reference about images**](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/img).

## How to complete this exercise

Look at the Workspace, below. You'll find the file index.html inside. Edit index.html in the workspace and watch your website change in the preview panel.

For this exercise, you need to create an image tag in the space provided and link it to the URL given in the HTML comment. Here's an example of what the HTML for an image element would look like:

<img src="http://somewebsite.com/image.jpg" alt="short description">

I want you to:

1. Create an <img> tag at the designated spot in the paragraph below.
2. Set the source to: http://udacity.github.io/fend/images/udacity.png
3. Set the tag's alt attribute to a description of the image (maybe something like, "Udacity logo").

You'll soon make a website that displays an image that is stored locally on your computer. In order to display a local image, you need to be able to write a **path**.

If there is a file called index.html in a directory and there is another directory called example/ in the same directory, you can access any files in example/ from index.html with the URL (path) example/filename.html, e.g. <a href="example/filename.html">Example Path</a>.

**Paths**

A path is a way of describing where a file is stored.

Think of it like this:

Anyone in the world can use the address 1600 Pennsylvania Ave NW, Washington DC, USA 20006 to find the White House. A street address is an absolute path to a location.

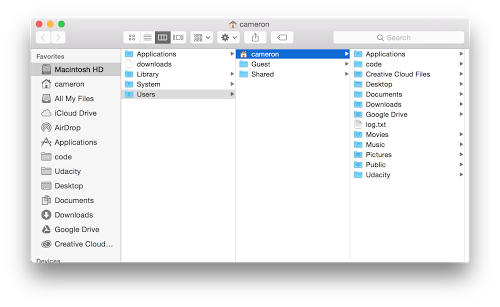
But, if you were at the [**Eisenhower Executive Office**](https://www.google.com/maps/place/Eisenhower+Executive+Office+Building/@38.8974712,-77.0390948,17z/data=!4m7!1m4!3m3!1s0x89b7b7bcdec17ee3:0xf920b148b3d45e45!2s1600+Pennsylvania+Ave+NW,+Washington,+DC+20500!3b1!3m1!1s0x0000000000000000:0x054470506cffbeb3), you could also use the phrase "next door" to find the White House. "Next door" is a relative path because it depends on your current location.

There are essentially two domains for paths that you'll need to consider as a web developer: paths to find files on your computer, **local** files, and paths to find files on other computers, **external** files.

**Local Paths**

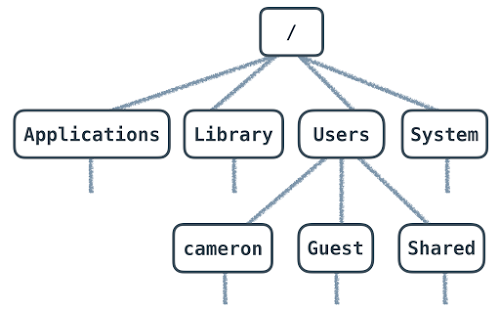
Computers have folders (also called "directories"). Operating systems like Windows, Mac and Linux organize *all* of your files into a tree of directories called a **file system**. There's a top-most directory, often called the **root**, that contains all of the other directories. Within the root, there are files and directories. Within those directories are more files and more directories. And within those directories are even more files and directories, and so on.

Compare this part of the file system on my computer:

**[[](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/73276037150923)](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/73276037150923)**

**[Local path directory structure screenshot](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/73276037150923)**

to a tree diagram showing the same directory structure:

**[[](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/73276037150923)](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/73276037150923)**

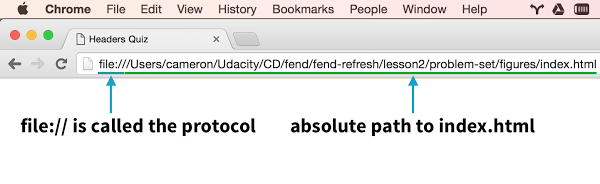
**[Tree diagram of directory structure](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/73276037150923)**

Every file has an address, which we call the "path." An absolute path is written in relation to the computer's root directory. For instance, a file in the Documents folder on a Mac has a path that looks like this:

/Users/cameron/Documents/file.txt

file.txt is stored inside Documents/. Users/, cameron/ and Documents/ are all names of directories. Documents/ lives inside cameron/ and cameron/ lives inside Users/. Users/ is inside the root directory, which is represented by the first /. The rest of the / are used to separate directories.

When you open an HTML file in your browser, you're seeing the absolute path to the file on your computer.

**[[](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/73276037150923)](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/73276037150923)**

**[Absolute Path to index.html](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/73276037150923)**

This URL will *only* work for you on your computer. As no one else has your file system, this URL is unique to your computer. If you want other people to be able to access it, then you need an...

**External Paths**

The process of loading a website from a URL like https://www.udacity.commimics opening an HTML file that you've written and saved to your computer. Every website starts with an HTML file. It just so happens that when you want to visit a website, the HTML file that you want to open lives on a different computer. The computer responsible for giving you a website's files is called a **server**.

Pointing your browser to https://www.udacity.com sends a request to Udacity's server for the HTML file (and others) that your computer needs to load the Udacity website. You can think of udacity.com as the root path of Udacity's server (computer) that anyone can access (the reality of the situation is actually much more complicated but the general idea is true). Unlike your personal computer (for now!), Udacity's servers run software that **expose** files to the web, which means that they make them available to anyone who wants them. Servers have an **external path** that anyone can access and is the reason why the web works.

Different websites are just different collections of files. Every website is really just a server (or many servers) with an external address, which we call a URL. Servers store files and send them to computers who request them (the requesting computers are called **clients**).

There are different **protocols** for serving files, the most common of which on the web are HTTP and HTTPS. When you open a file on your own computer, you're using the file protocol. You don't need to know much more about protocols for now, but if you're interested in learning (a lot!) more about HTTP, check out [**Networking for Web Developers**](https://www.udacity.com/course/networking-for-web-developers--ud256).

**Relative Paths**

The relative path is similar to the absolute path, but it describes how to find a path to a file from a directory that is not the root directory. Like using the phrase "next door" to tell someone how to find the White House from the Eisenhower Executive Office, a relative path takes advantage of the location of one file to describe where another file can be found.

Relative paths work the same for both local and external paths. Let's break down two examples of absolute paths to see how relative paths work.

**External**:

<a href="http://labs.udacity.com/fend/example/hello-world.html">Hello, world!</a>

**Local**:

<a href="/Users/cameron/Udacity/etc/labs/fend/example/hello-world.html"> Hello, world!</a>

href is really just a path to a file.

Both examples are links to the same file using absolute paths, but the external example would work for anyone and only my computer can use the link in the local example.

Pay attention to the fend/example/hello-world.html portion of both paths - they mean the same thing.

Imagine that you are editing /Users/cameron/Udacity/etc/labs/fend/test.html. test.html can reference hello-world.html by describing how to get from it's location in fend/ to hello-world.html. The relative path would look like:

example/hello-world.html

This relative path takes advantage of the fact that test.html and example/ are in the same directory.

But what if I'm editing a file in /Users/cameron/Udacity/etc/labs/ and I want to write a path to hello-world.html? In that case, the relative path would be:

fend/example/hello-world.html

Now that I'm in labs/, not fend/, I have to include fend/ in a relative path to hello-world.html.

To finish this off, let's imagine there are two files:

http://labs.udacity.com/science/sciences.html

and

http://labs.udacity.com/science/physics/relativity.html

In order to write a relative path from sciences.html to relativity.html, I only need to include the part of the path that describes how to get from science/ to relativity.html:

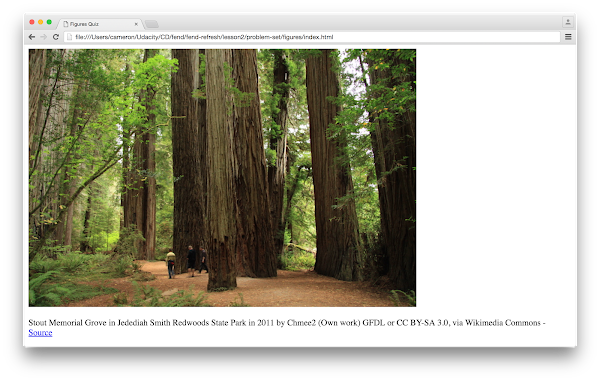
<a href="physics/relativity.html">Einstein's Special Relativity</a>

And that's it! Now it's time to apply your new skills.

## Figures

It's important to respect copyright and attribution on the web. When you use someone else's work, you need to give the author credit. With that in mind, for this exercise you're going to create an image with a caption underneath. I want you to give credit to the image's photographer by attributing them in the image's caption and providing a link to the source material.

This is what it should look like in your browser:

**[[](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/1fa1ca87-51c7-428e-9438-508c08f1d1a0)](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/1fa1ca87-51c7-428e-9438-508c08f1d1a0)**

Notice, in the image above, that there is a link to the source in the caption below the image.

### How to Complete this Exercise

For this exercise, you need to edit index.html so that the caption displays below the image. I want you to use a **relative** path to the image, which is going to be in the same directory as index.html. Remember, a relative path points to the file **from the current directory** - it does not start with the root. (Your relative path should not start with /, file://, C:\, or http://).

I want you to:

1. Create a <figure> element
2. Add an <img> element whose src attribute points to the provided image (look in the Workspace's sidebar for the "redwoods\_state\_park.jpg" image!)
3. Add an alt attribute to the image
4. Add a <figcaption> element with the text for the image's caption
5. Change the [**Markdown**](https://daringfireball.net/projects/markdown/) code to a real HTML link

Hint: If you get stuck, I recommend checking out the [**figure element on MDN**](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/figure).

## Mockup to Website

It's common for web developers to work with designers who focus on creating user interfaces and user experiences. Designers use software like Adobe Photoshop to mock up - draw - websites. The mockups that they create are usually just images of websites with some annotations and descriptions.

As a web developer, one of the tasks you might be asked to do is take a mockup created by a visual designer and translate it into a live website. I use the word "translate" because the process of going from mockup to website is similar to the process of translating between natural languages. Just as you can create the same meaning using different words and phrases with a natural language, you can create the same website design using different HTML elements.

I want you to practice the process of going from a mockup to a website now! Here is a website mockup (note: I zoomed in for the screen shot):

**[[](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/458dd2c0-4065-4cdf-b6a8-0cbdbb10a699)](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/7222405183/concepts/458dd2c0-4065-4cdf-b6a8-0cbdbb10a699)**

There are many ways to turn this mockup into website. The best end result isn't perfect HTML - it's getting your site to look the way it's supposed to. When you're done, compare your website to the mockup. You'll know you've finished this exercise when your site looks the same :)

### How to Complete this Exercise

I want you to recreate your website to look identical to the mockup. Start by taking a few minutes to analyze the mockup. What text is changed? What images are there? While doing this, I want you to practice indenting children elements. I'll show you how I indented my HTML in the solution.

This mockup was taken from [**this Udacity blog post**](http://blog.udacity.com/2016/01/hottest-jobs-in-2016-2-web-developer.html).

I want you to:

1. Position the text content to match the mockup
2. Style the content to match the mockup
3. Add any images that are missing (don't forget to include the alt attribute!)

# Exercise Overview

Hey there, web developer! Congratulations on making it this far! Here's your next exercise: **translating a mockup to HTML**. This is similar to the exercise from earlier in the problem set, except this article is going to be a bit more complicated. We'll be giving you a mockup of a website (and a copy of the text in it). It's going to be your job to recreate it with HTML.

You're going to see some formatting you haven't encountered before, so you'll need to use a resource, like the [**MDN element reference**](https://developer.mozilla.org/en-US/docs/Web/HTML/Element), to research new elements.

# Instructions

1. Download and unzip mockup-to-article.zip in the [**Supporting Materials section**](https://classroom.udacity.com/nanodegrees/nd001/parts/aabd87fc-6edf-4818-a911-15dfa9cabcb8/modules/4d4626ff-de39-4af7-9cf1-a26774ee022c/lessons/6c4b9e47-433b-4743-ab69-07f139b69404/concepts/744c1e15-7bd3-450d-b7d8-3a4dd635dca6#supporting-materials), below. You'll find index.html, the article mockup image (blog-mockup.pdf) and a file called reflections.txt inside.
2. Use what you've learned about web development so far to edit index.html so that it looks **exactly** like the mockup image. You will need to use new elements, which means that you'll need to [**research**](https://developer.mozilla.org/en-US/docs/Web/HTML/Element) and experiment! (Hint: look up "superscript," "horizontal rule," and "strikethrough."). Note: You can recreate the mockup without using a single <br> tag! You don't need to worry about line breaks. The text should naturally wrap depending on how wide the window is.
3. When you've finished building the article, take a look at the reflection questions below. You've learned a lot about web development so far. Take a moment to write down your thoughts about web development.

# Supporting Materials

* [**mockup-to-article.zip**](https://d17h27t6h515a5.cloudfront.net/topher/2016/December/58501f5b_mockup-to-article/mockup-to-article.zip)

**Note:** In previous versions of this Nanodegree program, this exercise was required for graduation. In the current version, this exercise is not submitted for review.